

2.2 Boolean Expressions

Name: _____ Class: _____ Date: _____

Total: 13 marks

KEY WORDS boolean expression • relational operators

Objective

Build the skills to answer exam questions on **Boolean expressions**.

You must be able to:

- write a **Boolean expression** that evaluates to **true** or **false**
- use the **relational operators** `<`, `<=`, `>`, `>=`, `==`, `!=`
- understand that `==` tests equality while `=` performs assignment

1 Key knowledge and worked examples

Study these first. Each one shows the method for a question type used later.

■ Boolean expressions

A Boolean expression evaluates to **true** or **false**, built with **relational operators**:

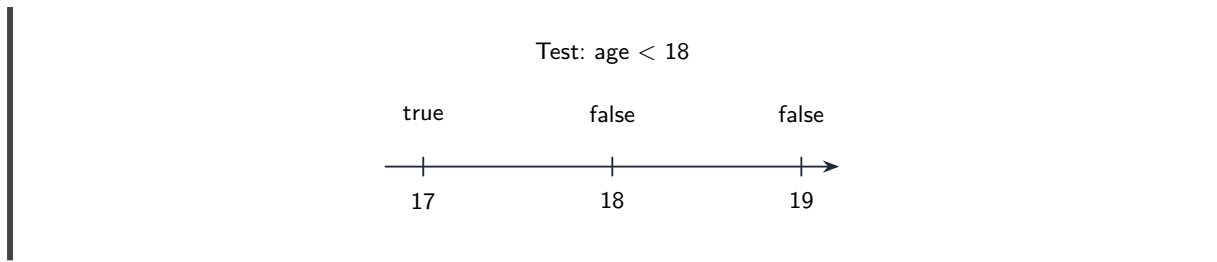
- `5 == 5` is **true** (equality)
- `3 != 4` is **true** (not equal)
- `7 >= 7` is **true**

■ `==` vs `=`

`==` **tests** equality; `=` **assigns** a value. Confusing them is a common bug.

■ Worked application: Test both sides of the boundary

An original Java rule grants a discount when an integer `age` is below 18. The expression `age < 18` is true for 17 and false for 18; `age <= 18` would incorrectly include the boundary. With `int age = 18;`, the statement `boolean eligible = age < 18;` assigns the result **false** to `eligible`. The expression `age == 18` instead tests equality and gives **true**. Testing 17, 18 and 19 catches a boundary error that testing only a young and an old age may miss. Equality tests do not change the stored age.



2 Practice

2.1 State what a Boolean expression evaluates to. [1]

2.2 State the difference between `==` and `=`. [2]

2.3 Evaluate `7 >= 7`. [1]

3 Exam-style questions

3.1 Which operator tests equality? [1]

☐

A `=`

B `==`

C `!=`

D `<=`

3.2 `4 != 4` evaluates to [1]

☐

A true

B false

C 4

D an error

3.3 Given `int x = 6, y = 9;`:

(a) evaluate `x < y`. [1]

(b) evaluate `x == y`. [1]

(c) evaluate `x != y`. [1]

3.4 A Java programme defines `int score = 60;` and `boolean pass = score >= 60;`. State `pass`, compare with the result of `score > 60`, and give two neighbouring integer inputs useful for checking this boundary. [4]

Solutions

Each answer shows the steps, then the **result**.

2.1

- either `true` or `false`

2.2

- `==` tests whether two values are equal; `=` assigns a value to a variable

2.3

- `true`

3.1 B

- `==`.

3.2 B

- `false`.

3.3

(a)

- `true`

(b)

- `false`

(c)

- `true`

3.4

`pass` is `true`; `score > 60` is `false`; inputs 59 and 61; these lie immediately below and above the boundary, checking exclusion and inclusion respectively.